LISTING OF CLAIMS

1.   (Currently Amended) A method for optimizing an executable comprising:

migrating a plurality of objects from a first memory to a second memory**, wherein migrating includes**

> **determining whether each object of the plurality of objects are accessible from a plurality of processors,**
>
> **determining an equivalence set of aliased objects in the plurality of objects,**
>
> **determining objects of the plurality of objects eligible for migration,**
>
> **changing residence of the objects determined to be eligible for migration, and**
>
> **changing accesses of the objects having their residence changed**;

determining alignment of the migrated plurality of objects; and

eliminating redundant initialization code of the plurality of objects.

2.   (Original) The method for optimizing an executable of claim 1, wherein the plurality of objects are variables.

3.   (Cancelled).

4.   (Original) The method for optimizing an executable of claim 1, the determining alignment further comprising:

analyzing the migrated objects by forward disjunctive dataflow analysis;

determining a minimum alignment necessary for each migrated object; and

setting the minimum alignment necessary for each migrated object.

5.   (Original) The method for optimizing an executable of claim 1, wherein the first memory is an external memory and the second memory comprises a plurality of indexed registers residing in a microengine.

6.   (Currently Amended) A processing device comprising:

an optimizer to **determine whether each variable of a plurality of variables are accessible from at least two network processors, determine an equivalence set of aliased variables in the plurality of variables, change location of the variables that are determined to be eligible for migration, and** migrate [[a]] **the** plurality of objects from an external memory of a network processing device to a plurality of registers coupled to a processor, the optimizer further to align and eliminate redundant initialization code of the plurality of objects.

7.   (Original) The processing device of claim 6, wherein the plurality of registers are indexed.

8.   (Original) The processing device of claim 6, wherein the plurality of objects are variables.

9.   (Original) The processing device of claim 6, wherein the migrated plurality of objects are not shared by the processor and at least one other processor.

10.   (Original) The processing device of claim 6, wherein the network processing device is a router.

11.   (Currently Amended) An optimizer system for network processors comprising:

a processor,

a first memory coupled to the processor;

a display coupled to the processor; ~~and~~

a compiler to migrate a plurality of objects from a second memory to a plurality of

indexed registers in a network processor, the compiler further to align and eliminate redundant

initialization code of the plurality of objects**;**

a first determiner to determine whether each object of the plurality of objects are

accessible from a plurality of processors in a network device;

a second determiner to determine an equivalence set of aliased objects in the

plurality of objects;

a third determiner to determine objects of the plurality of objects eligible for

migration;

a migrator to change residence of the objects determined to be eligible for

migration; and

an accessor to change accesses of the objects having their residence changed.

12.   (Cancelled).

13.	(Currently Amended) The optimizer system for network processors of claim [[12]]11, wherein the second memory is external to the plurality of processors.

14.	(Original) The optimizer system for network processors of claim 11, wherein the plurality of objects are variables.

15.	(Original) The optimizer system for network processors of claim 11, wherein the second memory is external to the plurality of indexed registers.

16.	(Currently Amended) A ~~machine-accessible~~ **machine-readable storage** medium containing instructions that, when executed, cause a machine to:

determine whether each variable of a plurality of variables are accessible from at least two network processors;

determine an equivalence set of aliased variables in the plurality of variables;

change location of the variables that are determined to be eligible for migration;

migrate [[a]] **the** plurality of variables from a first memory to a plurality of indexed registers;

align the migrated plurality of variables; and

eliminate redundant initializations to a base address register.

17.	(Cancelled).

18. (Currently Amended) The ~~machine-accessible~~ **machine-readable storage** medium of claim 16, further comprising instructions that, when executed, cause a machine to:

analyze the migrated variables by forward disjunctive dataflow analysis; and

determine a minimum alignment necessary for each migrated variable.

19. (Currently Amended) The ~~machine-accessible~~ **machine-readable storage** medium of claim 16, further comprising instructions that, when executed, cause a machine to:

set the minimum alignment necessary for each migrated variable.

20. (Currently Amended) The ~~machine-accessible~~ **machine-readable storage** medium of claim 16, further comprising instructions that, when executed, cause a machine to:

compile source code to migrate the plurality of variables from the first memory to the plurality of indexed registers.